MAVEN POM

POM stands for *Project Object Model* . It is fundamental Unit of Work in Maven. It is an XML file. It always resides in the base directory of the project as pom.xml.

The POM contains information about the project and various configuration details used by Maven to build the project(s).

POM also contains the goals and plugins. While executing a task or goal, Maven looks for the POM in the current directory. It reads the POM, gets the needed configuration information, then executes the goal. Some of the configuration that can be specified in the POM are following:

- project dependencies
- plugins
- goals
- build profiles
- project version
- developers
- mailing list

Before creating a POM, we should first decide the project **group** (groupId), its **name**(artifactId) and its version as these attributes help in uniquely identifying the project in repository.

Example POM

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
   http://maven.apache.org/xsd/maven-4.0.0.xsd">
   <modelVersion>4.0.0</modelVersion>

   <groupId>com.companyname.project-group</groupId>
   <artifactId&gtproject</artifactId>
   <version>1.0</version>

</project>
```

It should be noted that there should be a single POM file for each project.

- All POM files require the **project** element and three mandatory fields: **groupId, artifactId,version.**
- Projects notation in repository is **groupId:artifactId:version.**
- Root element of POM.xml is **project** and it has three major sub-nodes :

Node	Description
groupId	This is an Id of project's group. This is generally unique amongst an organization or a project. For example, a banking group com.company.bank has all bank related projects.

artifactId	This is an Id of the project.This is generally name of the project. For example, consumer-banking. Along with the groupId, the artifactId defines the artifact's location within the repository.
version	This is the version of the project.Along with the groupId, It is used within an artifact's repository to separate versions from each other. For example: *com.company.bank:consumer-banking:1.0* *com.company.bank:consumer-banking:1.1.*

Super POM

All POMs inherit from a parent (despite explicitly defined or not). This base POM is known as the **Super POM**, and contains values inherited by default.

Maven use the effective pom (configuration from super pom plus project configuration) to execute relevant goal. It helps developer to specify minimum configuration details in his/her pom.xml. Although configurations can be overridden easily.

An easy way to look at the default configurations of the super POM is by running the following command: **mvn help:effective-pom**

Create a pom.xml in any directory on your computer.Use the content of above mentioned example pom.

In example below, We've created a pom.xml in C:\MVN\project folder.

Now open command console, go the folder containing pom.xml and execute the following **mvn** command.

```
C:\MVN\project>mvn help:effective-pom
```

Maven will start processing and display the effective-pom.

```
[INFO] Scanning for projects...
[INFO] Searching repository for plugin with prefix: 'help'.
[INFO] ------------------------------------------------------------------------
[INFO] Building Unnamed - com.companyname.project-group:project-name:jar:1.0
[INFO]    task-segment: [help:effective-pom] (aggregator-style)
[INFO] ------------------------------------------------------------------------
[INFO] [help:effective-pom {execution: default-cli}]
[INFO]

.....

[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESSFUL
[INFO] ------------------------------------------------------------------------
[INFO] Total time: < 1 second
[INFO] Finished at: Thu Jul 05 11:41:51 IST 2012
[INFO] Final Memory: 6M/15M
[INFO] ------------------------------------------------------------------------
```

Effective POM displayed as result in console, after inheritance, interpolation, and profiles are applied.

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- ============================================================== -->
<!--                                                                -->
<!-- Generated by Maven Help Plugin on 2012-07-05T11:41:51          -->
<!-- See: http://maven.apache.org/plugins/maven-help-plugin/        -->
```

```
<!--                                                                        -->
<!-- ====================================================================== -->

<!-- ====================================================================== -->
<!--                                                                        -->
<!-- Effective POM for project                                              -->
<!-- 'com.companyname.project-group:project-name:jar:1.0'                   -->
<!--                                                                        -->
<!-- ====================================================================== -->

<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/
2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 h
ttp://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.companyname.project-group</groupId>
  <artifactId>project</artifactId>
  <version>1.0</version>
  <build>
    <sourceDirectory>C:\MVN\project\src\main\java</sourceDirectory>
    <scriptSourceDirectory>src/main/scripts</scriptSourceDirectory>
    <testSourceDirectory>C:\MVN\project\src\test\java</testSourceDirectory>
    <outputDirectory>C:\MVN\project\target\classes</outputDirectory>
    <testOutputDirectory>C:\MVN\project\target\test-classes</testOutputDirectory>
    <resources>
      <resource>
        <mergeId>resource-0</mergeId>
        <directory>C:\MVN\project\src\main\resources</directory>
      </resource>
    </resources>
    <testResources>
      <testResource>
        <mergeId>resource-1</mergeId>
        <directory>C:\MVN\project\src\test\resources</directory>
      </testResource>
    </testResources>
    <directory>C:\MVN\project\target</directory>
    <finalName>project-1.0</finalName>
    <pluginManagement>
      <plugins>
        <plugin>
          <artifactId>maven-antrun-plugin</artifactId>
          <version>1.3</version>
        </plugin>
        <plugin>
          <artifactId>maven-assembly-plugin</artifactId>
          <version>2.2-beta-2</version>
        </plugin>
        <plugin>
          <artifactId>maven-clean-plugin</artifactId>
          <version>2.2</version>
        </plugin>
        <plugin>
          <artifactId>maven-compiler-plugin</artifactId>
          <version>2.0.2</version>
        </plugin>
        <plugin>
          <artifactId>maven-dependency-plugin</artifactId>
          <version>2.0</version>
        </plugin>
        <plugin>
          <artifactId>maven-deploy-plugin</artifactId>
          <version>2.4</version>
        </plugin>
        <plugin>
          <artifactId>maven-ear-plugin</artifactId>
          <version>2.3.1</version>
        </plugin>
        <plugin>
          <artifactId>maven-ejb-plugin</artifactId>
          <version>2.1</version>
        </plugin>
        <plugin>
          <artifactId>maven-install-plugin</artifactId>
```

```
        <version>2.2</version>
      </plugin>
      <plugin>
        <artifactId>maven-jar-plugin</artifactId>
        <version>2.2</version>
      </plugin>
      <plugin>
        <artifactId>maven-javadoc-plugin</artifactId>
        <version>2.5</version>
      </plugin>
      <plugin>
        <artifactId>maven-plugin-plugin</artifactId>
        <version>2.4.3</version>
      </plugin>
      <plugin>
        <artifactId>maven-rar-plugin</artifactId>
        <version>2.2</version>
      </plugin>
      <plugin>
        <artifactId>maven-release-plugin</artifactId>
        <version>2.0-beta-8</version>
      </plugin>
      <plugin>
        <artifactId>maven-resources-plugin</artifactId>
        <version>2.3</version>
      </plugin>
      <plugin>
        <artifactId>maven-site-plugin</artifactId>
        <version>2.0-beta-7</version>
      </plugin>
      <plugin>
        <artifactId>maven-source-plugin</artifactId>
        <version>2.0.4</version>
      </plugin>
      <plugin>
        <artifactId>maven-surefire-plugin</artifactId>
        <version>2.4.3</version>
      </plugin>
      <plugin>
        <artifactId>maven-war-plugin</artifactId>
        <version>2.1-alpha-2</version>
      </plugin>
    </plugins>
  </pluginManagement>
  <plugins>
    <plugin>
      <artifactId>maven-help-plugin</artifactId>
      <version>2.1.1</version>
    </plugin>
  </plugins>
</build>
<repositories>
  <repository>
    <snapshots>
      <enabled>false</enabled>
    </snapshots>
    <id>central</id>
    <name>Maven Repository Switchboard</name>
    <url>http://repo1.maven.org/maven2</url>
  </repository>
</repositories>
<pluginRepositories>
  <pluginRepository>
    <releases>
      <updatePolicy>never</updatePolicy>
    </releases>
    <snapshots>
      <enabled>false</enabled>
    </snapshots>
    <id>central</id>
    <name>Maven Plugin Repository</name>
    <url>http://repo1.maven.org/maven2</url>
  </pluginRepository>
```

```
  </pluginRepositories>
  <reporting>
    <outputDirectory>C:\MVN\project\target/site</outputDirectory>
  </reporting>
</project>
```

In above pom.xml , you can see the default project source folders structure,output directory, plug-ins required, repositories, reporting directory which Maven will be using while executing the desired goals.

Maven pom.xml is also not required to be written manually.

Maven provides numerous archetype plugins to create projects which in order create the project structure and pom.xml

Details are mentioned in Maven plug-ins and Maven Creating Project Sections